6H−2

軽量フォーマルメソッドを利用したコンポーネントソフトウェア向け検証ツールに関する研究*

橋本 吉治[†], 松本 充広[†§], 二木 厚吉[†]

北陸先端科学技術大学院大学 情報科学研究科

e-mail:{hashimo, mitihiro, kokichi}@jaist.ac.jp

1. はじめに

生産性向上の期待からコンポーネントソフトウェア開発(以下，CBDと略す)の普及が始まっている．CBDは，コンポーネントを部品として組み合わせ，ソフトウェアを開発する手法である．しかし，現在のCBDの問題点は，基本的な機能を提供するコンポーネントを組み合わせて作成したソフトウェアの振る舞い（以下，コンポーネントソフトウェアの振る舞いと略す）と，顧客が要求するソフトウェアの振る舞いが一致する保証がない点である．

本論文では，コンポーネントソフトウェアの信頼性向上に寄与するために，上で述べたコンポーネントソフトウェアの振る舞いと顧客が要求するソフトウェアの振る舞いが一致することを検証するツールを提案する．なお，この検証ツールは，軽量フォーマルメソッド[1] を理論的基礎とする．

2. コンポーネントソフトウェア

現在のコンポーネントソフトウェア開発では，ソフトウェア作成ツールの上でコンポーネント同士を結びつける手法が多く用いられている．しかし，この手法では，コンポーネントソフトウェアの振る舞いと顧客が要求するソフトウェアの振る舞いが一致する保証はない．

筆者らは，UML，コンポーネントソフトウェア開発手法，軽量フォーマルメソッドを使用したコンポーネント管理システム[‡]の研究[1]を進めている．このシステムは，対象ドメインを設定し，そのドメインのビジネスモデル，ドメイン用（仕様付き）コンポーネントライブラリ，ドメインに含まれるソフトウェアの仕様を管理する．そして，ビジネスモデル，仕様間の一貫性検証を行い，ソフトウェアの仕様からのソフトウェアの生成を行う．更に生成したソフトウェアの配布も行う．

本研究では，上記のコンポーネント管理システムのうち，ビジネスモデル，仕様間の一貫性検証を実現するシステムを議論する．

3. 検証システムの概要

3. 1. 検証システムの概要

本研究では，先に述べたコンポーネント管理システムにおける自動検証部の構築を行う．その際使用する軽量フォーマルメソッドはUMLを使用したカタリシス法[2]ベースである．そのため，分析，設計レベルのシームレスな統合が可能となり，詳細化検証が可能となる．また，技術者に理解されやすい．具体的な検証内容は次の通りである．(1)ビジネスモデルとソフトウェアの仕様との間の一貫性検証，(2)抽象レベルの仕様と具体レベルの仕様との間の一貫性検証，(3)ビジネスモデル，ソフトウェア仕様で記述した不変条件の検証．システムへの入力にはUML図を使用する．UML図を記述するガイドラインは，筆者らが研究開発したガイドライン[1](以下，UMLガイドラインと略す)を使用する．意味の付与，検証を行う仕様言語は，代数仕様言語CafeOBJ[3]を使用する．

---

* A component software verification tool using a lightweight formal method

† Yoshiharu HASHIMOTO, Michihiro MATSUMOTO , Kokichi FUTATSUGI Graduate School of Information Science, Japan Advanced Institute of Science and Technology

§ On leave from First division, PFU Limited

‡ このシステムは情報処理振興事業協会(IPA)の委託により財団法人ソフトウェア工学研究財団(RISE)が実施している平成13年度「高度情報化支援ソフトウェアシーズ育成事業」にて，(株)PFUが北陸先端大の協力のもと，作成しているものである．なお，第二筆者は，このプロジェクトの研究開発リーダである．

3. 2. 検証例

本節では，前節(2)の一貫性検証に必要なズームイン・ズームアウトの説明をする．カタリシス法におけるアクション，オブジェクトは，それぞれUMLのユースケース，アクターと対応する．カタリシス法では，属性値の変化でアクションの効果を記述する．図１に属性値の変化例を示す．図１の例では，hashimoの属性値であるp-possessがアクションの事前と事後で変化することが分かる．つまり，アクションの事前でのp-possseeの値はfalse（oilを持っていない）であるが，アクションの事後での値はtrue（oilを持つ）となる．

アクション，オブジェクトは分割できる．図２にシーケンス図を用いたズームイン・ズームアウト例を示す．アクションbuyは，makeorder, notifyorder, deliver, pay, というアクションに分解できる．また，オブジェクトVendorはSales, Distribution, Accountsというオブジェクトに分解できる．そして，makeorder, notifyorder, deliver, payを組み合わせたときのアクションの効果が最終的にズームアウトしたbuyアクションの効果と等しいことを軽量フォーマルメソッドを使用して検証する．軽量フォーマルメソッドとは，問題の対象領域を固定して，その領域の性質を用いて検証を可能な限り容易にするフォーマルメソッドのことである．

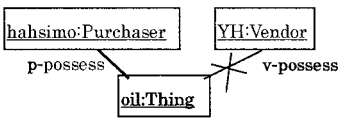

図２．属性値の変化例

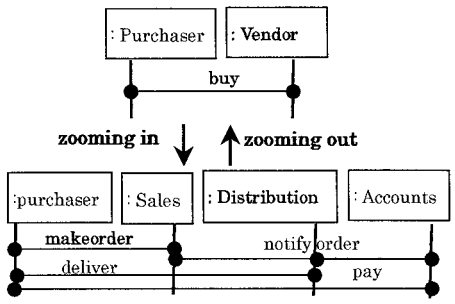

図２．　ズームイン・ズームアウト例（シーケンス図）

3. 3. 検証システムの処理フロー

図３に検証システムの処理フローを示す．　本検証システムでは，入力はXMI表現のUML図とする．処理１で，入力されたUML図から，UML図名などの必要な情報を抽出する．抽出された情報は中間データとして保存され，処理２で，(1)CafeOBJの仕様と(2)仕様を検証するためのスクリプトに変換する．上記の(1)，(2)はCafeOBJ検証系上で検証され，最後に出力として検証結果が表示される．

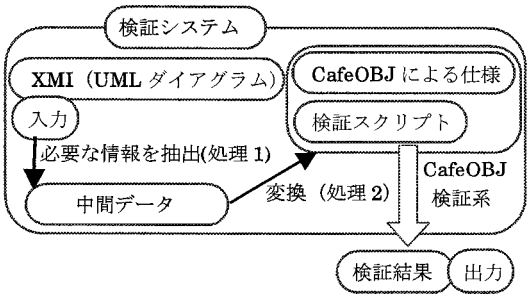

図３．　検証ツールの処理フロー

4．関連研究

軽量フォーマルメソッドの研究としてはAlloy[4]がある．Alloyの目的は読み書きが容易で，記述した仕様の解析の自動化が可能な最小のモデル言語を提供することである．Alloyもクラス図制約の検証手法を提供するが，ズームイン・ズームアウトの一貫性検証手法は提供しない．

5．まとめ

本研究では，軽量フォーマルメソッドに基づいたUML図の一貫性，普遍条件を自動検証するシステムを提案した．今後の予定として，このシステムを実装する．

参考文献

[1]松本充広，二木厚吉:「コンポーネントソフトウェア開発用軽量フォーマルメソッド」，信学技法，2001

[2]http://www.catalysys.org

[3]R．Diaconescu and K.Futatsugi：“CafeOBJ Report”,World Scientific,1998.

[4]D.Jackson:"Alloy:A-Lightweight-Object-Modelling-Notation",http://sdg.lcs.mit.edu/~dnj/pub, 2000.